



# **CURSO**

## **TECNICO EM INFORMATICA**

### **DISCIPLINA**

#### **LINGUAGEM DE PROGRAMAÇÃO-I**

# A Linguagem Java

## INTRODUÇÃO À LINGUAGEM JAVA

- Linguagem de programação divulgada pela Sun Microsystems em 1995:
  - ✓ início do desenvolvimento: 1991;
  - ✓ líder da equipa: James Gosling da Sun Microsystems;
  - ✓ a partir de 2009, pertence à Oracle;
- Orientada a objectos(OO):
  - ✓
  - ✓ os programas são **classes**;
  - ✓ Noção de **objecto**é fundamental;  
Propriedades OO: abstração, encapsulamento, herança e polimorfismo.

# INTRODUÇÃO À LINGUAGEM JAVA

## PORQUE JAVA É IMPORTANTE

- Java possui portabilidade, confiança e segurança;
- Essas são características que Java apresenta, tornam-se requisitos essenciais para uma linguagem de programação.
- Java oferece várias camadas de controle de segurança para a proteção contra código malicioso;
- Além de possuir um compilador altamente rigoroso que evita erros básicos de programação;
- Ajudando a manter a integridade do **software**, um dos conceitos essenciais em Governança.
- Além dessas características, há um diferencial que deve ser considerado: Java é uma solução gratuita.

# A Linguagem Java

## ➤ Identificadores

- ✓ Identificadores em Java são usados para dar nomes:
  - A variáveis, rótulos, classes, pacotes, interfaces, enumerações, métodos, campos, parâmetros formais e constantes.
- ✓ Todo identificador começa com uma letra:
  - podendo ser seguido de uma sequencia de qualquer tamanho contendo letras, dígitos, ou o caractere de sublinhado ( ).

# A Linguagem Java

- ~~Não haverá~~ Orientação na formação de identificadores em relação ao **programa**, mas, com o objetivo de facilitar a legibilidade de programas, a seguinte padronização recomendada é:
  - ✓ nomes de classes e de interfaces devem iniciar-se com letra maiúscula;
  - ✓ nomes de constantes devem ser escritos com todas as letras em maiúsculo;
  - ✓ nomes de outros elementos de um programa devem iniciar-se com letra minúscula.
- Os seguintes identificadores são usados para designar palavras chave da linguagem Java e, por isto, não podem ser usados para outros fins:
  - ✓ **Em declarações:** boolean, byte, char, double, final, float, int, long, private, protected, public, short, static, transient, void, volatile;
  - ✓ **Em expressões:** null, new, this, super, true, false;
  - ✓ **Em comandos de seleção:** if, else, switch, case, break, default.

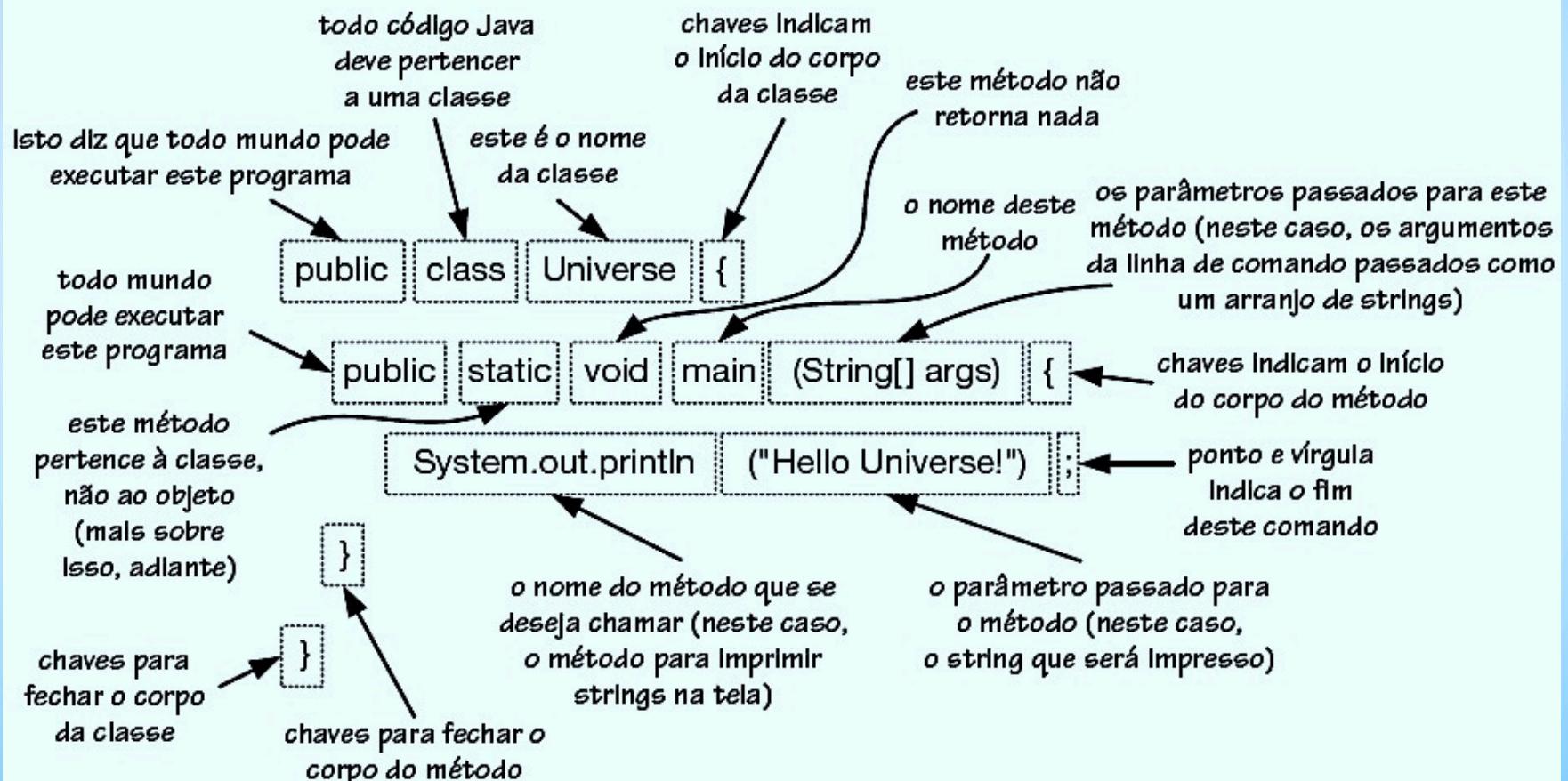
# A Linguagem Java

## CLASSE E TIPOS DE OBJETOS

- Classe é um conjunto de objetos com características comuns:
- ✓ Uma classe é como um modelo para a criação de objetos, que tem as mesmas características da classe à qual pertence.
- A definição de classe normalmente inclui:
  - Modificador de acesso: *public, private*
  - A palavra-chave *class*
  - Campos das instâncias
  - Constructores
  - Métodos das instâncias
  - Campos da classe
  - Métodos da classe

# A LINGUAGEM JAVA

## CLASSE E TIPOS DE OBJETOS



# A LINGUAGEM JAVA

## CLASSES E TIPOS DE OBJETOS

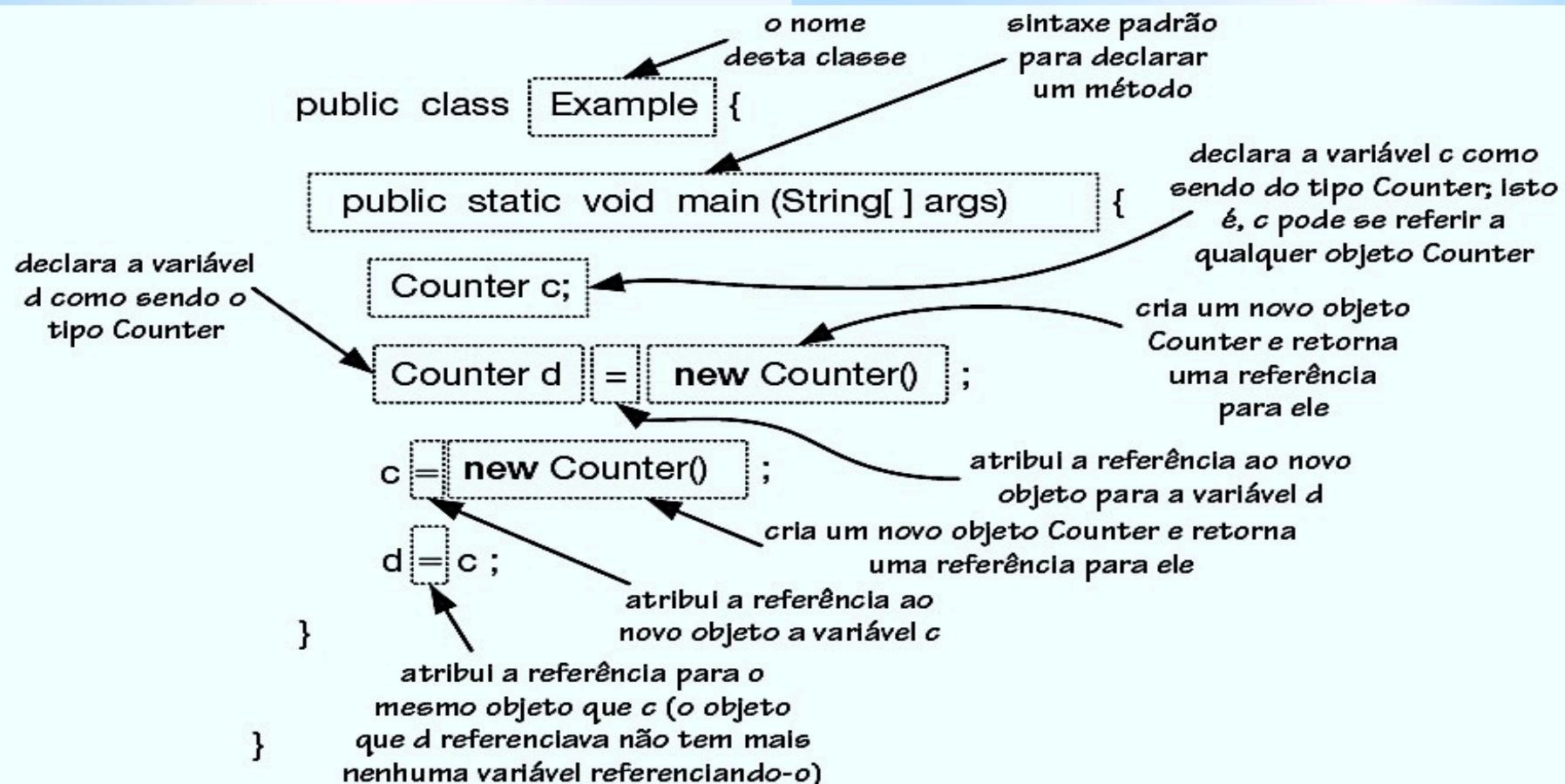
- Os principais “atores” em um programa Java são os *objetos*;
- ✓ Os objetos armazenam dados e fornecem os métodos para acessar e modificar esses dados;
- ✓ Todo objeto é instância de uma **classe** que define o *tipo* do objeto, bem como os tipos de operações que executa;
- ✓ Dados de objetos Java são armazenados em **variáveis de instância** (também chamadas de **campos**).
- ✓ As variáveis de instância podem ser de tipos básicos (tais como inteiros, números de ponto flutuante ou booleanos) ou podem se referir a objetos de outras classes;
- ✓ As operações que podem atuar sobre os dados e que expressam as “mensagens” às quais os objetos respondem são chamadas de *métodos*;
- ✓ E estes consistem em construtores, subprogramas e funções. Eles definem o comportamento dos objetos daquela classe

# A LINGUAGEM JAVA

## CLASSE E TIPOS DE OBJETOS

### ➤ Objetos:

- ✓ Em Java, um objeto novo é criado a partir de uma classe usando-se o operador new;
- ✓ O operador **new** cria um novo objeto a partir de uma classe especificada e retorna uma referência para este objeto.



# A LINGUAGEM JAVA

## CLASSES E TIPOS DE OBJETOS

- ✓ A chamada do operador new sobre um tipo de classe faz com que ocorram três eventos:
  - Um novo objeto é dinamicamente alocado na memória, e todas as variáveis de instância são inicializadas com seus valores padrão;
  - O construtor para o novo objeto é chamado com os parâmetros especificados.
  - Depois de o construtor retornar, o operador new retorna uma referência (isto é, um endereço de memória) para o novo objeto recém-criado

# A LINGUAGEM JAVA

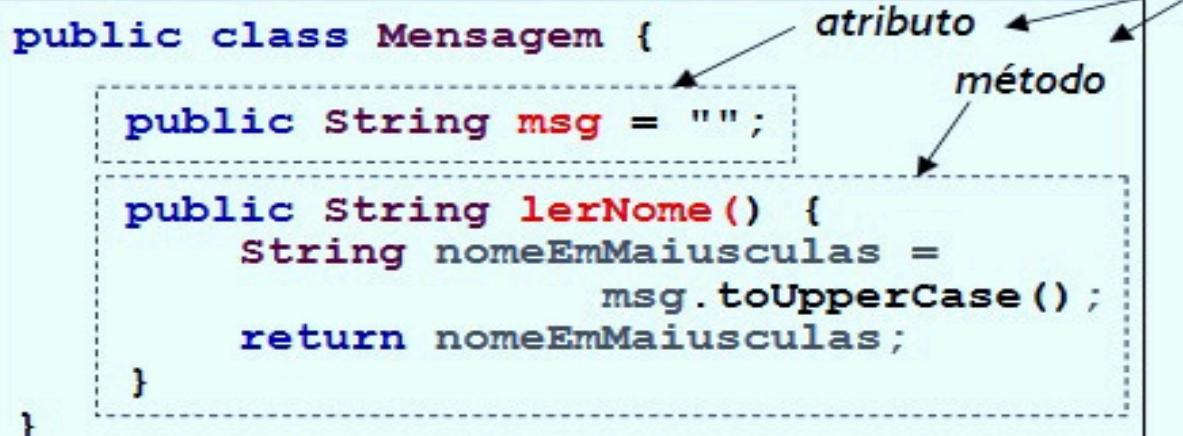
## CLASSE E TIPOS DE OBJETOS

Uma classe define um tipo de dados

- Esta classe representa objetos que guardam um texto (tipo `String`) em um atributo (`msg`) publicamente acessível.
- Além de guardar um `String`, retorna o texto em caixa-alta através do método `lerNome()`.

Definição da classe (tipo) `Mensagem` em Java

```
public class Mensagem {  
    public String msg = "";  
  
    public String lerNome() {  
        String nomeEmMaiusculas =  
            msg.toUpperCase();  
        return nomeEmMaiusculas;  
    }  
}
```



**Membros da classe.** Outras classes podem acessá-los, se declarados como "public", usando o operador ponto "."

Representação  
em UML

Mensagem
+msg: String
+lerNome(): String

Esta é a interface pública da classe. É só isto que interessa a quem vai usá-la. Os detalhes (código) estão encapsulados.

# A LINGUAGEM JAVA

## CLASSE E TIPOS DE OBJETOS

### ■ Declaração do método *main()*

```
public static void main(String[] args)
```

*modificadores*

*tipo de dados  
retornado*

*nome*

*variável local ao método que  
contém valor passado na chamada*

*tipo de dados aceito  
como argumento*

### ■ O método *main()* é chamado pelo interpretador Java, automaticamente

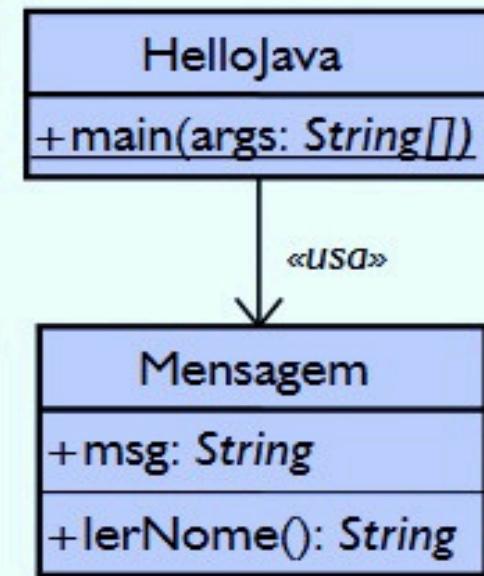
- Deve ter **sempre** a assinatura acima

### ■ O argumento é um **vetor** formado por textos passados na linha de comando:

```
> java NomeDaClasse Um "Dois Tres" Quatro
```

↑           ↑           ↑  
*args[0]*   *args[1]*   *args[2]*

Dependência entre as duas classes (HelloJava tem referência para Mensagem)



# A LINGUAGEM JAVA

## O CÓDIGO FONTE

- Como qualquer outra linguagem de programação, Java é usada para criar aplicações de computador;

```
/***
 * Instituto de Software do Ceará - INSOFT
 * XI Semana tecnológica de férias
 * Primeiro programa - escrever a mensagem alô mundo na tela.
 */
public class AloMundo
{
    static public void main(String[] args)
    {
        System.out.println("Alô mundo");
    }
}
```

- As primeiras 5 linhas representam um bloco de comentário, que tem por objetivo identificar a função do programa, seu autor, versão, etc.

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

Bloco:

Bloco é uma estrutura que permite reunir um grupo de comandos e um escopo local de declarações;

Um comando bloco tem um dos formatos:

- { declarações e comandos }
- { comandos }

Deve ser usado sempre que se quiser codificar vários comandos onde apenas um for permitido ou se desejar delimitar o escopo de certas declarações.

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

- A linha seguinte (public class AloMundo) declara uma classe chamada AloMundo.
- Após compilado, esse código gerará um arquivo **AloMundo**;
- Um código fonte em Java pode descrever mais de uma classe;
- Após a compilação, cada descrição de classe gerará um arquivo ;
- Observe que pode haver no máximo um **classe public** dentro de cada código-fonte Java;
- Caso você inadvertidamente declare mais de uma classe como **public** dentro de um código-fonte Java, ocorrerá um erro de compilação.

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

- **Métodos:** O conjunto de funcionalidades da classe;
- ✓ Para cada método, especifica-se sua assinatura, composta por:
  - Nome:** umidade identificadora para o método;
  - Tipo:** quando o método tem um valor de retorno, o tipo desse valor
  - Lista de argumentos:** quando o método recebe parâmetros para sua execução, o tipo e um identificador para cada parâmetro.
  - Visibilidade:** como para atributos, define o quanto visível é um método a partir de objetos de outras classes.
- ✓ A seguir encontra-se a declaração do método inicial de um programa Java.  
Todo programa Java começa a ser executado pelo método **main**:  
`static public void main(String[] args)`

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

### ➤ **Static:**

- ✓ É usado para a criação de uma variável que poderá ser acessada por todas as instâncias de objetos desta classe como uma variável comum;

Ex:Neste caso que podem ser executados apenas no contexto da **classe AloMundo**.

### ➤ **public:**

- ✓ Uma declaração com o modificador public pode ser acessada de qualquer lugar e por qualquer entidade que possa visualizar a classe a que ela pertence;
- ✓ No exemplo acima o método **main** opdeser executado por qualquer processo ativo no sistema operacional, incluindo o interpretador Java.

### ➤ **Void:**

- ✓ Indica a ausência de um tipo de dado. Serve para indicar um método e não retorna nenhum dado.
- ✓ Quando um tipo de retorno é declarado como **void**, significa que o método não retorna nenhum valor.

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

- **String[] args:**
- ✓ Um array de objetos do tipo String, que serve para armazenar a lista de argumentos digitados na linha de comando após o nome da classe a ser executada:

```
c:\alo>java AlloMundo nome numero ...
```

```
c:\alo>_
```

- ✓ Esses argumentos são acessados pelo índice do vetor args, ou seja, args[0] = nome, args[1] = numero, etc.

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

Teste:

```
public class TesteContemAluno {  
    public static void main(String[] args) {  
        Aluno a1 = new Aluno();  
        Aluno a2 = new Aluno();  
  
        a1.setNome("Rafael");  
        a2.setNome("Paulo");  
  
        Vetor lista = new Vetor();  
  
        lista.adiciona(a1);  
        lista.adiciona(a2);  
  
        System.out.println(lista.contem(a1));  
  
        System.out.println(lista.contem(a2));  
  
        Aluno aluno = new Aluno();  
        aluno.setNome("Ana");  
  
        System.out.println(lista.contem(aluno));  
    }  
}
```

Teste:

```
public class TesteTamanhoLista {  
    public static void main(String[] args) {  
        Aluno a1 = new Aluno();  
        Aluno a2 = new Aluno();  
        Aluno a3 = new Aluno();  
  
        a1.setNome("Rafael");  
        a2.setNome("Paulo");  
  
        Vetor lista = new Vetor();  
  
        lista.adiciona(a1);  
        lista.adiciona(a2);  
  
        System.out.println(lista.tamanho());  
  
        lista.adiciona(a3);  
  
        System.out.println(lista.tamanho());  
    }  
}
```

# A LINGUAGEM JAVA

## O CÓDIGO FONTE

- **Comentários em Java:**
- ✓ Para fazer um comentário em java, você pode usar o // para comentar até o final da linha, ou então.
  - usar o : /\* \*/ para comentar o que estiver entre eles.

EX:

```
/* comentário daqui,  
ate aqui */  
  
// uma linha de comentário sobre a idade  
int idade;
```

# A LINGUAGEM JAVA

## TIPOS DE DADOS

- Os tipos de dados de Java são classificados em primitivos ou referencias:
- ✓ **Os tipos primitivos são:**
  - os tipos numéricos inteiros (byte, char, short, int, long);
  - os numéricos fracionários (float, double) e;
  - os booleanos (boolean (true e false));
  - Os tipos que denotam referencias comprehendem classes, interfaces e arranjos.
- ✓ Variáveis de tipo primitivo obedecem ao modelo denominado semântica de valor, e contem diretamente um valor do tipo Declarado
- ✓ Variáveis declaradas do tipo referência seguem o modelo semântica de referência, assim, armazenam, não um valor do tipo declarado, mas o endereço do objeto que contem um valor do tipo declarado.

# A LINGUAGEM JAVA

## TIPOS DE DADOS

### Tipos primitivos

- **Tipos lógicos: boolean**
- ✓ Valores lógicos possuem dois estados, normalmente ditos **verdadeiro/falso, sim/não e ligado/ desligado.**
- ✓ Em Java um tipo lógico é definido pela palavra boolean, e pode assumir dois valores: true ou false.

```
// Exemplo de variável que suporta valores booleanos
boolean anoBissexto = false;
boolean anoPar = true;

// Apesar de uma variável poder ser declarada
// sem receber um valor, ela só poderá ser usada
// após a atribuição de algum valor a ela.
boolean valido;
```

# A LINGUAGEM JAVA

## TIPOS DE DADOS

### ➤ Tipos textuais: char e String:

- ✓ Caracteres simples são representados pelo tipo char;
- ✓ Um char representa um caracter , um número inteiro sem sinal de 16 bits, no intervalo de 0 até 2<sup>16</sup>-1. O valor de um literal char deve ser delimitado por aspas simples:

```
// Exemplo de representação de caracteres UNICODE
char primeiraLetra = 'a';
char tabulacao = '\t';

// Código UNICODE para o caractere de interrogação
char unicode = '\u0A02';

// Lembre-se: Uma variável só poderá
// ser manipulada após receber um valor.
char inutil;      // variável sem utilidade neste momento
inutil = '@';     // variável útil a partir de agora
```

- ✓ Palavras são representadas por uma sequência de dados do tipo **char**, agrupadas em um tipo especial de dados: a classe String.

# A LINGUAGEM JAVA

## TIPOS DE DADOS

- **Tipos numéricos inteiros: byte, short, inte long**
- ✓ Existem quatro tipos primitivos de números em Java. Além disso, os valores numéricos podem ser representados de forma decimal, octal ou hexadecimal:

```
// Valores inteiros representáveis pelos tipos
// numéricos em Java:
byte a      = 127;           // -27 ... 27 -1
short b     = 32767;          // -215 ... 215 -1
int c       = 2147483647;    // -231 ... 231 -1
long d      = 9223372036854775807L; // -263 ... 263 -1
```

# A LINGUAGEM JAVA

## TIPOS DE DADOS

- **Tipos numéricos de ponto flutuante: float e double**
- ✓ Um valor fracionário pode ser representado em Java através dos tipos float e double;
- ✓ A diferença entre esses dois tipos é o tamanho:
  - float 32 bits;
  - double 64 bits.

```
// Representação de valores numéricos de ponto flutuante
float pi    = 3.141516;
float taxa = 6.02E23;
double valor= 123.4E+306D;
```

- ✓ todo valor numérico de ponto flutuante é considerado do tipo *double*, a menos que o programador o declare explicitamente como *float*.

# A LINGUAGEM JAVA

## TIPOS DE DADOS

TABELA DOS TIPOS DE DADOS PRIMITIVOS

Tipos	Primitivo	Valores possíveis		Valor Padrão	Tamanho	Exemplo
		Menor	Maior			
Inteiro	byte	-128	127	0	8 bits	byte ex1 = (byte)1;
	short	-32768	32767	0	16 bits	short ex2 = (short)1;
	int	-2.147.483.648	2.147.483.647	0	32 bits	int ex3 = 1;
	long	-9.223.372.036.854.770.000	9.223.372.036.854.770.000	0	64 bits	long ex4 = 1l;
Ponto Flutuante	float	-1,4024E-37	3.40282347E + 38	0	32 bits	float ex5 = 5.50f;
	double	-4,94E-307	1.79769313486231570E + 308	0	64 bits	double ex6 = 10.20d; ou double ex6 = 10.20;
Caractere	char	0	65535	\0	16 bits	char ex7 = 194; ou char ex8 = 'a';
Booleano	boolean	false	true	false	1 bit	boolean ex9 = true;

# A LINGUAGEM JAVA

## OPERADORES LÓGICOS E ARITIMÉTICOS

- Operadores Aritmética: Os operadores aritméticos são **operadores binários**,
- ✓ ou seja, funcionam com dois operandos. Por exemplo, a expressão “ $a + 1$ ” contém o operador binário “+” (**mais**) e os dois operandos “ $a$ ” e “ $1$ ”.

Operação	Operador	Expressão algébrica	Expressão Java
Adição	+	$a + 1$	$a + 1$
Subtração	-	$b - 2$	$b - 2$
Multiplicação	*	$c m$	$c * m$
Divisão	/	$d / e$	$d / e$
Resto	%	$f \text{ mod } g$	$f \% g$

Operador	Operação	Ordem de avaliação(precedência)
* / %	Multiplicação Divisão Resto	Avaliado primeiro. Se houver vários operadores desse tipo serão avaliados da esquerda para a direita
+ -	Adição Subtração	Avaliado em seguida. Se houver vários operadores desse tipo, serão avaliados da esquerda para a direita.
=	Atribuição	Avaliado por último

# A LINGUAGEM JAVA

## OPERADORES LÓGICOS E ARITIMÉTICOS

Operador de igualdade	Operador de igualdade	Exemplo de condição em Java	Significado da condição em Java
<b>Operadores de igualdade</b>			
=	==	x == y	x é igual a y
?	!=	x != y	x é diferente de y
<b>Operadores relacionais</b>			
>	>	x > y	x é maior que y
<	<	x < y	x é menor que y
>_	>=	x >= y	x é maior que ou igual a y
<_	<=	x <= y	x é menor que ou igual a y

# A LINGUAGEM JAVA

## OPERADORES LÓGICOS E RELACIONAIS

- Os operadores relacionais e lógicos são utilizados em testes e condições de entrada em um fluxo do programa. Abaixo estão todos eles relacionados.

Op	Nome	Uso	Descrição
>	Maior que	$x > y$	$x$ maior que $y$ .
$\geq$	Maior ou igual a	$x \geq y$	$x$ maior ou igual a $y$ .
<	Menor que	$x < y$	$x$ menor que $y$
$\leq$	Menor ou igual a	$x \leq y$	$x$ menor ou igual a $y$ .
$=\!=$	Igual a	$x == y$	$x$ igual a $y$ .
$!=$	Diferente de	$x != y$	$x$ diferente de $y$ .
!	NÃO lógico (NOT)	$!y$	contrário de $y$ .
$\&\&$	E lógico (AND)	$x \&\& y$	$x$ e $y$ .
$\ $	OU lógico (OR)	$x \  y$	$x$ ou $y$ .
instanceof	Verif Instância	$x instanceof X$	$x$ é instância da classe $X$